# Systematische Datenbank-Performance-optimierung als Projektaufgabe

Jörg Stahnke, Ppi

Performance wird oft nur als Projektrisiko betrachtet. Wenn Probleme auftreten, werden diese „ad hoc" gelöst. Der Artikel zeigt, wie man Performanceprobleme in einer Oracle-Datenbank durch ein systematisches Vorgehen von Anfang an vermeiden kann.

Sowohl für agile Projekte als auch bei Wasserfallprojekten gibt es häufig pro Sprint bzw. Release den in *Abbildung 1* dargestellten Ablauf:

Im Rahmen der Konzeption konzentriert man sich auf fachliche Anforderungen. Im Entwicklertest wird nur mit geringen Datenmengen gearbeitet. Erst im Abnahme-/ Integrationstest werden größere Datenmengen getestet und „ganz plötzlich" stellt man Performanceprobleme fest. Manchmal treten diese Probleme auch erst im Produktivbetrieb auf. Die Performanceprobleme werden erst sehr spät erkannt und man hat nur noch wenig Zeit. Deshalb bleiben oft nur zwei Varianten. Entweder man verschiebt den Termin der Produktivnahme oder man behebt in einer Nacht-und-Nebel-Aktion schnell die schlimmsten Symptome. Beide Alternativen sind nicht empfehlenswert und führen zu weiteren ungeplanten Kosten.

Schlechte Performance kann zu einem Hamsterradeffekt im Team führen.

Die Anwendung hat eine schlechte Performance. Dadurch sind Testzyklen sehr aufwendig. In der Konsequenz werden weniger Tests durchgeführt. Es dauert lange, bis der Entwickler weiß, ob sein Code das gewünschte Ergebnis liefert. Dadurch steigt die Fehlerrate. Durch weniger fachliche Testzyklen werden mehr Fehler übersehen und erst im Produktivbetrieb fallen diese auf. Dann müssen Hotfixes erstellt und im schlimmsten Fall zusätzliche Bereinigungsprogramme für fehlerhaft verarbeitete Daten geschrieben werden. Der Aufwand für den Tagesbetrieb steigt. In der Konsequenz hat das Team noch weniger Zeit, sich um Performancefragen zu kümmern, und es entsteht ein sich selbst verstärkender Hamsterradeffekt *(siehe Abbildung 2)*. Im Extremfall kann dies zur völligen Handlungsunfähigkeit führen.

## Performance als Projektaufgabe

Den „Hamsterradeffekt" muss man auf jeden Fall vermeiden. Deshalb darf man

Performance nicht als Projektrisiko, um das man sich bei Bedarf kümmert, betrachten. Stattdessen muss Performance als Projektaufgabe von Anfang an geplant werden. Dies umfasst:

- Know-how
- Zeit
- Budget
- Mitarbeiter

Nach meiner Erfahrung ist die Suche nach der genauen Fehlerursache der schwierigste Schritt im Rahmen der Performanceoptimierung. Leider wird über Performance häufig in größeren Runden oder sogar speziellen Task-Force-Gruppen sehr ineffizient diskutiert *(siehe Abbildung 3)*.

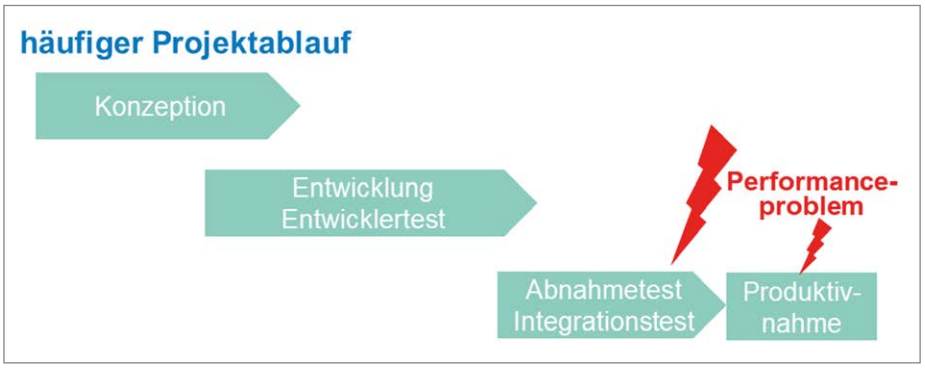


*Abbildung 1: Häufiger Projektablauf (Quelle: Jörg Stahnke)*

Eine auf Vermutungen basierende Diskussion ist ineffizient und kostet viel Zeit. Verantwortlichkeiten werden zwischen Entwicklern, Datenbankadministratoren, Betriebssystembetreuern, dem Netzwerk, der Hardware und anderen hin- und hergeschoben. Es gilt das Motto „Schuld sind immer die anderen". Meine Erfahrung ist außerdem, dass das Problem fast immer an einer völlig unerwarteten Stelle liegt. Dieser Effekt ist dadurch zu erklären, dass die Teammitglieder über erwartete performancekritische Stellen des Programms intensiv nachgedacht haben und deshalb hier eine sehr gute Lösung umgesetzt wurde. An Stellen, für die man keine Performanceprobleme erwartet, erfolgt dies nicht.

Deshalb hat sich für mich folgendes Vorgehen bewährt:

1. detaillierte Messungen durchführen
2. Fakten ermitteln
3. danach eine Diskussion führen

Diskussionen, die auf Messwerten und Fakten basieren, sind wesentlich zielgerichteter. Man kann über konkrete Tatsachen sprechen und findet schneller Lösungen *(siehe Abbildung 3)*.

## Messung mithilfe eines Black-Box-Verfahrens

Die Oracle-Datenbank bietet die Möglichkeit, sich mithilfe der SQL-Tuning-Pa-

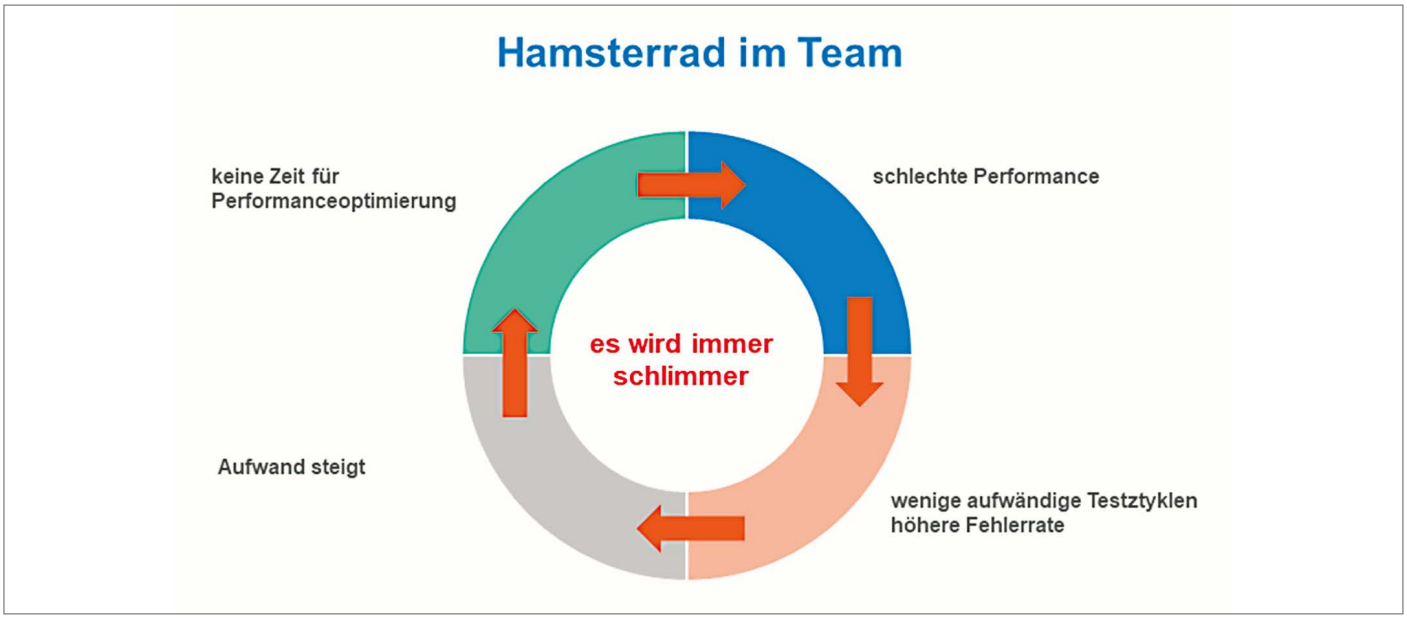


*Abbildung 2: Hamsterrad im Team (Quelle: Jörg Stahnke)*

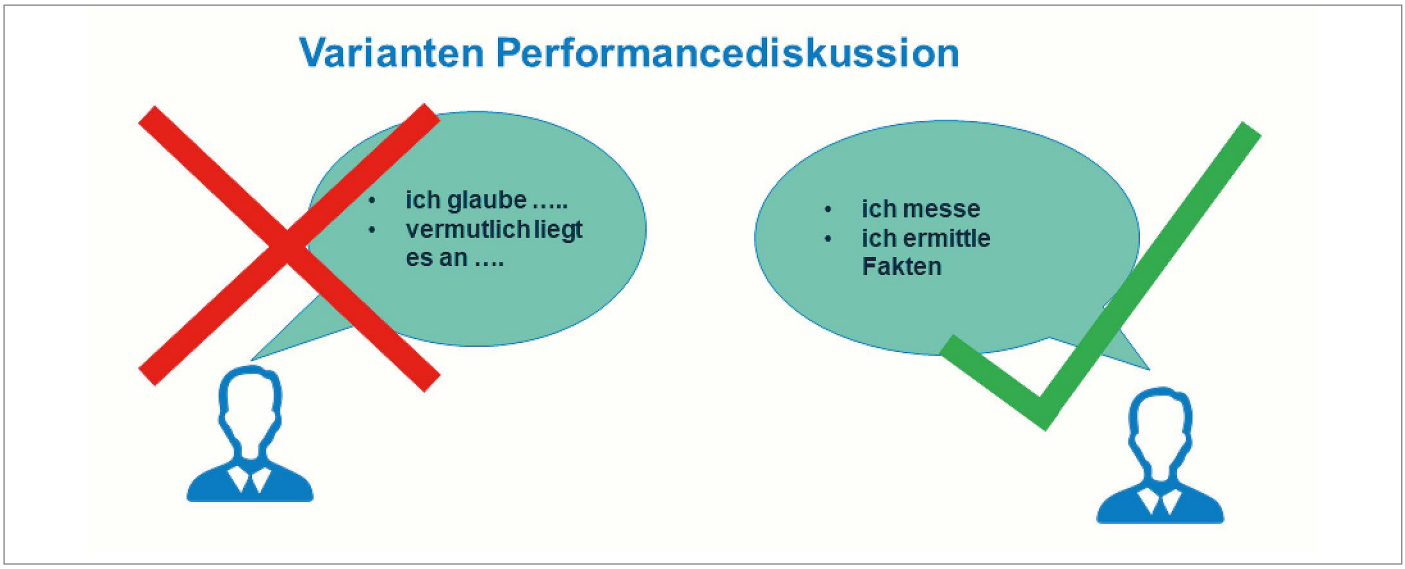


*Abbildung 3: Varianten der Performancediskussion (Quelle: Jörg Stahnke)*

ckages zwischen Applikation und Datenbank „einzuklinken" und den Workload dort detailliert zu erfassen. Dies ist in *Abbildung 4* dargestellt. Dazu muss man nur wissen, mit welchem Datenbankbenutzer sich die Anwendung an der Datenbank anmeldet. Der Ablauf ist wie folgt:

- Start der SQL-Tuning-Funktionalitäten
- Start der zu testenden Anwendung

Der Start des SQL-Tuning *[1]* erfolgt automatisiert per Skript.

Häufig sind für diese Messungen keine gesonderten Tests notwendig. Man misst einfach Tests, die gemäß Projektplanung durchgeführt werden. Geplante fachliche Tests oder regelmäßige Entwicklertests werden durch Performancemessungen begleitet. So entsteht kein Mehraufwand.

Die Performancemesswerte werden de facto als „Abfallprodukt" gewonnen. Die Messwerte stehen in Form von Datenbanktabellen zur Verfügung und können leicht ausgewertet werden.

## Gewonnene Messwerte

Das erste wichtige Ergebnis liefert den Anteil der Datenbank an der Gesamtlaufzeit. Wenn von einer Stunde Laufzeit nur 5 Sekunden auf die Datenbank entfallen, kann man die Datenbankanalyse sofort beenden und sich auf andere Komponenten konzentrieren. Gerade in Data-Warehouse-Anwendungen hat die Datenbank häufig jedoch einen entscheidenden Anteil an der Gesamtlaufzeit. In den meisten Fällen sind es überraschenderweise nur 3 verschiedene SQL-Statements, die bis zu 90% der Gesamtlaufzeit ausmachen. Daher hat man sofort eine klare Aussage darüber, auf welche Bereiche sich die Optimierung konzentrieren muss.

Für jedes einzelne SQL-Statement liegen unter anderem folgende Messwerte vor:

- Wie oft wurde es ausgeführt?
- Wie viele Zeilen wurden geliefert?
- Wie lange hat es gedauert?
- Nach welchem Plan wurde es ausgeführt?
- Welche Teilschritte der Ausführung kosten die meiste Zeit?
- Gab es Wartezustände?
- Werden zu viele oder zu wenige Indizes verwendet?

Diese in *Abbildung 5* dargestellten Messwerte liefern ein klares Bild, wo das Problem ist und welche Ursache dahinter steckt.

Beispiele wären:

- ein SQL dauert nur 0,1 Millisekunden, wird aber 5 Millionen mal pro Stunde ausgeführt

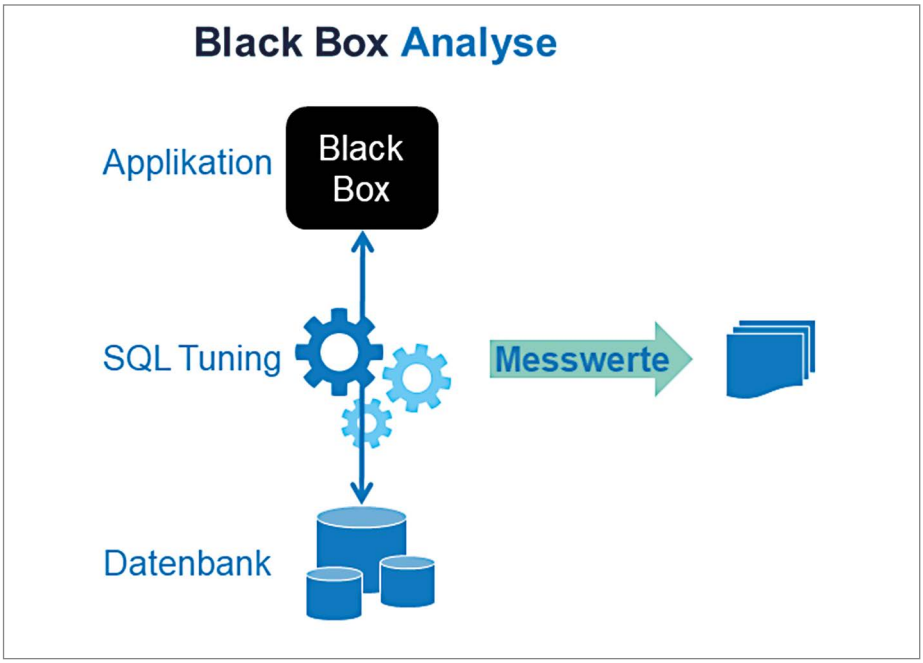


*Abbildung 4: Ablauf der Blackbox-Messung (Quelle: Jörg Stahnke)*

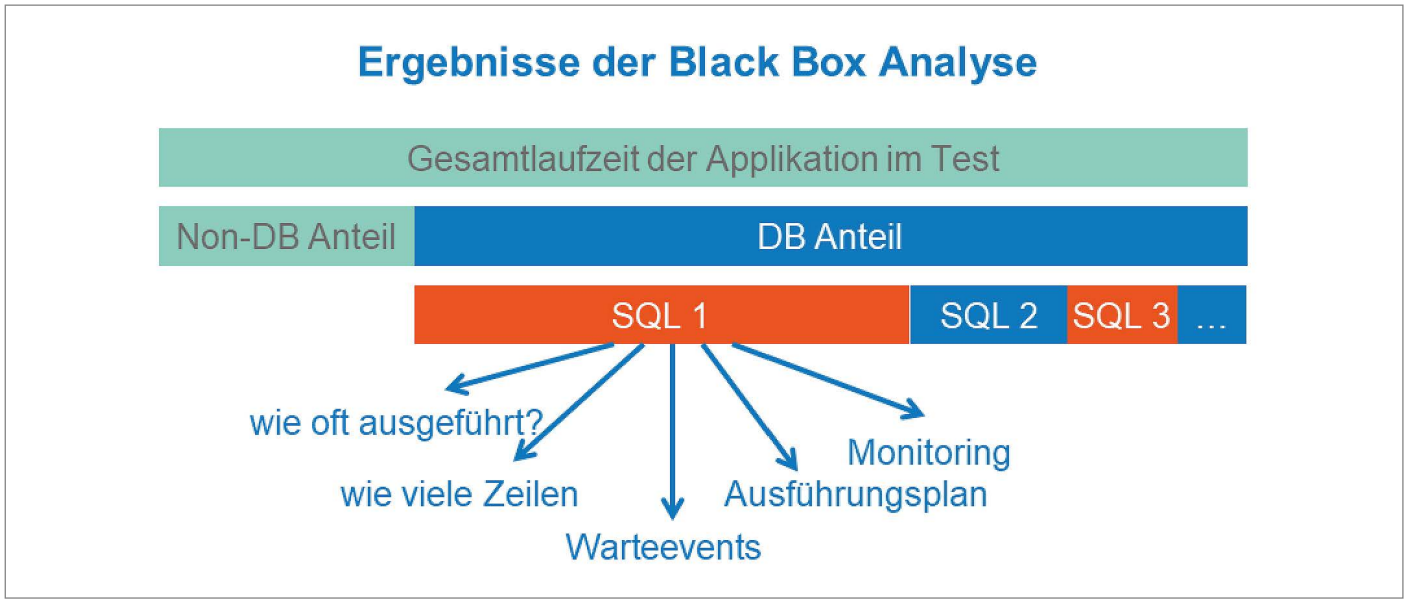


*Abbildung 5: Übersicht über die Messwerte (Quelle: Jörg Stahnke)*

- ein Index wird nicht verwendet, weil bei der Abfrage durch einen Tippfehler die Datentypen falsch übergeben wurden
- das Partition Pruning funktioniert nicht wie erwartet
- der Oracle Optimizer schätzt die Anzahl der Ergebniszeilen falsch ein und verwendet dadurch einen falschen Ausführungsplan
- wider Erwarten wird kein Parallel Query verwendet
- die Datenbank führt extrem viele I/O-Operationen auf den Plattensystemen aus
- umfangreiche (evtl. unnötige?) Sortiervorgänge
- häufige Wartezustände durch konkurrierende Datenzugriffe

Die Messwerte weisen also sehr detailliert auf die Ursachen der Performanceprobleme hin. Dadurch kann sofort eine Lösungssuche auf Detailebene erfolgen. Diese Arbeitsweise ist extrem zielgerichtet und effizient.

Der Aufwand für die Umsetzung der Lösungen kann dabei stark schwanken. Im einfachsten Fall wird das Problem durch eine Typ-inkonsistente Parameterübergabe aufgrund eines Tippfehlers verursacht. Dies kann innerhalb einer Minute korrigiert werden. Im schlimmsten Fall muss die Anwendungsarchitektur geändert werden, da es sonst zu massiven Wartezuständen auf der Datenbank kommt.

## Kosten- und Nutzen-Betrachtung

Wenn Sie im Projekt systematisch Performance planen, müssen Sie dafür Zeit, Mitarbeiter und Budget vorsehen. Die Erarbeitung und Umsetzung von Lösungen erfordern Aufwand. Diesen Ausgaben stehen allerdings Einnahmen gegenüber. Ungeplante „Ad-hoc"-Optimierungen zum Projektende entfallen. Sie können effizienter entwickeln und testen, da alle Tests schneller sind. Entwickler arbeiten schneller und fehlerfreier. Die Frage „Hat mein Code das gewünschte Ergebnis?" wird in kürzerer Zeit beantwortet.

Netto entstehen daher keine Kosten. Dafür gibt es folgende Vorteile:

- Sie agieren, statt auf Probleme zu reagieren
- hohe Effizienz der Entwicklung
- zufriedenere Anwender
- entspannteres Arbeiten

In der Praxis fallen im Projekt die Kosten für die Performanceoptimierung immer an. Aus meiner Erfahrung hat man als Projektleiter im Prinzip nur zwei Wahlmöglichkeiten:

- Das Budget wird in einem geordneten Prozess geplant ausgegeben und die Applikation hat eine sehr gute Performance.
- Das Budget muss man zum Schluss in hektische Ad-hoc-Maßnahmen investieren und hat trotzdem nur eine teilweise optimierte Anwendung.

## Empfehlung

Implementieren Sie für jedes Release den in *Abbildung 6* dargestellten Projektablauf für die Performance-Optimierung.

Mit Start des Release wird die beschriebene Black-Box-Analyse durchgeführt. Für die drei SQL-Statements mit der höchsten Gesamtlaufzeit wird eine detaillierte Analyse durchgeführt. Mithilfe von Modifizierungen des SQL-Statements wird losgelöst von der eigentlichen Applikation durch Datenbankspezialisten ein Lösungsvorschlag erarbeitet. Dabei wird zielgerichtet je nach Messwert/Ursache des Problems gearbeitet.

Sowohl die Messwerte als auch die Ursachen und damit die Lösungsansätze können sehr vielfältig sein. Immer wieder gibt es Überraschungen.

Mögliche Lösungsansätze sind zum Beispiel:

- geänderte DB-Parameter
- mehr/weniger Indizes
- Partitionierung
- Komprimierung
- Parallel Query
- technische Änderung des SQL-Statements
- minimale fachliche Änderung des SQL-Statements

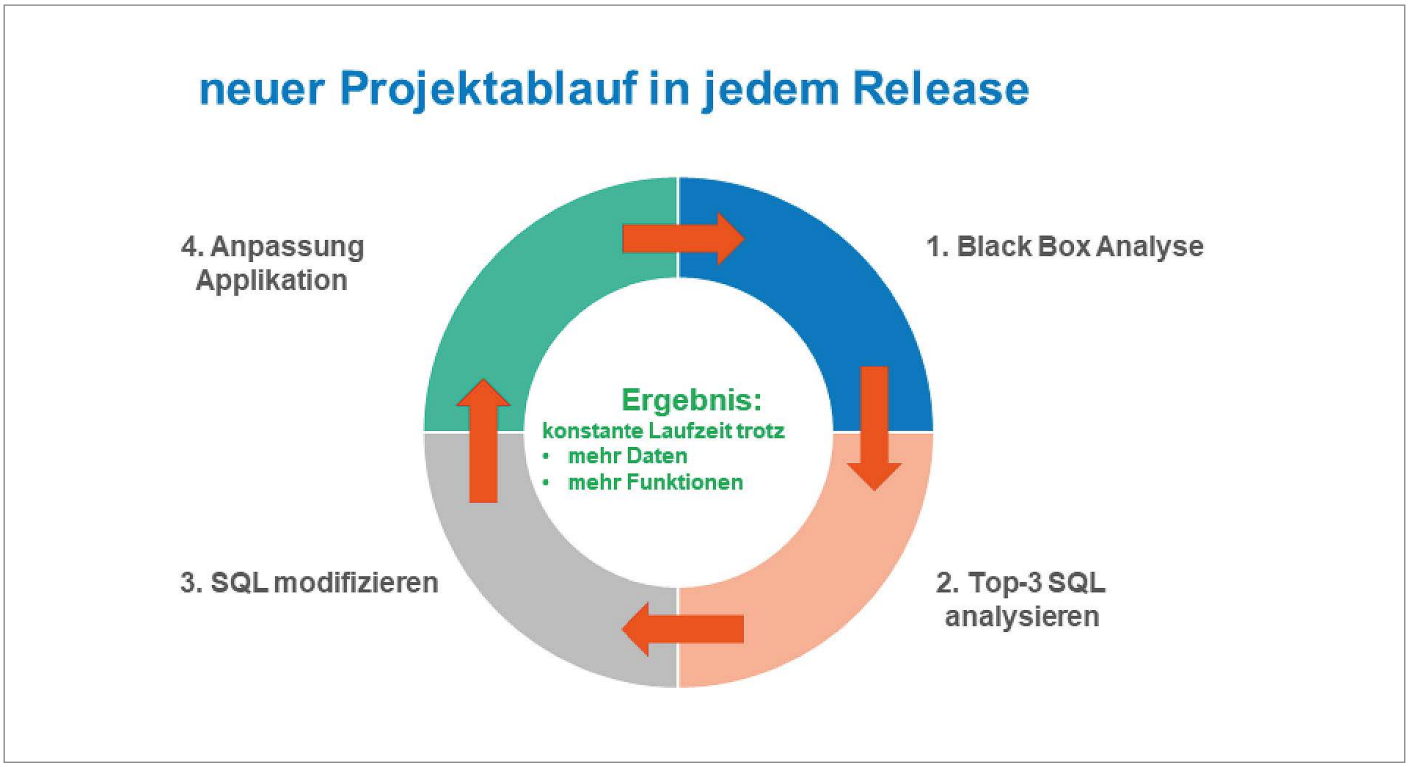


*Abbildung 6: Neuer Projektablauf (Quelle: Jörg Stahnke)*

Ziel ist es, eine Variante des SQL-Statements zu finden, die eine sehr gute Performance hat.

Danach wird mit den Anwendungsentwicklern diskutiert, wie dieses geänderte SQL-Statement in die Anwendung integriert werden kann. Von Antworten wie „Das geht nicht mit unserem Framework" darf man sich nicht entmutigen lassen. Häufig wäre die korrekte Antwort „Ich weiß noch nicht, wie dies mit unserem Framework geht" gewesen.

In Einzelfällen kann über Performance-kritische fachliche Anforderungen ohne großen Nutzen mit den Anwendern gesprochen werden. Beispielweise kann eine vollständige Protokollanzeige auf die Anzeige der letzten 2 Wochen reduziert werden, wenn nur die aktuellen Einträge fachlich interessant sind.

Im Ergebnis bleiben die Laufzeiten konstant, obwohl sich die Datenmenge im Data Warehouse durch Anbindung neuer Systeme erhöht hat und weitere fachliche Funktionalitäten ergänzt wurden.

## Fazit

Unerwartete Performanceprobleme in Oracle-Datenbanken kann man durch folgendes Vorgehen ausschließen:

1. Performance muss geplant werden
2. systematisch mit Messwerten/Fakten arbeiten
3. keine Vermutungen aufstellen
4. gezielt Know-how im Team aufbauen
5. SQL-Tuning aktiv nutzen
6. regelmäßige Optimierungen in jedem Release

Mit diesen Empfehlungen werden Performanceprobleme von Anfang an vermieden und müssen nicht gelöst werden.

## Quellen

[1] Der vollständige SQL Tuning Advisor erfordert eine zusätzliche Lizenzierung. Falls diese Lizenz nicht gekauft wurde, kann jedoch trotzdem eine Vielzahl von Messwerten lizenzfrei ermittelt werden.

## Über den Autor

Jörg Stahnke verfügt über eine mehr als 20-jährige Erfahrung im Bereich Data Warehouse. Seine Spezialgebiete sind dabei die Erstellung von Data-Warehouse-Architekturen sowie die Performanceoptimierung. Dabei legt er sehr viel Wert auf Architekturen, die eine effiziente Entwicklung und schnelle Änderungszyklen ermöglichen sowie potenzielle Fehlerquellen vermeiden. Bereits seit Beginn seiner Tätigkeit setzt er Generatoren ein, um Entwicklungsprozesse zu automatisieren. Seit 7 Jahren arbeitet er daher mit Data Vault.

Jörg Stahnke
joerg.stahnke@ppi.de